CS112 - Fall 2022

Optional Lab

Instructor: Paul Haskell

INTRODUCTION

In this lab, you will extend some of the software you developed last week. This will give you experience reviewing and improving your own software. In my personal experience, the best way to improve your software quality is the same as the best way to improve your English paper quality: read it again and again, and keep editing and improving.

Extensions to class Deck

What operations do we want from a deck of cards?

- Deal cards so we can play a card game
- Shuffle the deck, making it ready for a new game

In a future week, we will shuffle our card deck at random. But for now, implement a `shuffle()` method to `class Deck` that simply restores the deck to its original condition, with no cards dealt yet. After some or all the cards in the deck have been dealt, `shuffle()` should make every card in the deck available for dealing again. For this week, after `shuffle()`, the cards can be in whatever order you want.

You should already have a method in `class Deck` that sets the deck to its original condition: the constructor. You need to modify `class Deck` so this method can be called multiple times.

Also, implement a `deal()` method in `class Deck`:

- The lone argument to `deal()` is an `int`, saying how many cards to deal during this method call
- The return type of `deal()` is an array of cards, `Card[]`

You must extend `class Deck` so it keeps track of what cards have been dealt and what cards have not yet been dealt. It should be possible to call `deal()` multiple times without ever receiving the same card, until `shuffle()` is called.

If `deal()` is called requesting more cards than remain available in the deck, then `deal()` should return `null`. It should not return fewer cards than were requested.

Think about how to test your new `Deck`. What test cases must you try? (What cases do you think the grader will try?)

Finally, please modify your **MyCardDeck.java** program as follows: it should take one or more command line arguments. Each argument is either an integer or the word "**shuffle**". For each input integer, the program should deal that many cards and print them out on a single line. If the input is "**shuffle**", then the deck should shuffle itself.

- Remember that if a deal request is made that cannot be fully fulfilled, then the request should be ignored. The program should print "**null**" for this case. For example, if someone calls

```
java MyCardDeck   52   1
```
then the program should print:
```
<<a list of all the cards in the deck>>
null
```

Reminder

Put useful comments into your code. Design and organize your code so that it is easy for others to understand.

This work is optional and for fun, but I will take a look at your code and will give you comments. Please do not change the files in your **Lab06** directory for this work! Put all your files for this optional work in **Lab07** and let me know via email.